Université Bordeaux 1 — Projet
Master 1 CSI — 2005–2006

Réseau : paire à paire

Modalités :

- Le projet est à faire par groupe de 4 personnes.
- Le projet devra être rendu au plus tard le vendredi 09 décembre à 17h00.
- La note de contrôle continu sera égale à la note de projet.
- Toute fraude détectée, notamment copié/collé, sera sanctionnée par une note <=5 : pensez à interdire le droit d'accès en lecture de vos documents... (sous unix, chmod og-r monProjet).

A rendre :

- Les codes sources, abondamment documentés, et archivés. L'archive ne doit contenir ni binaire, ni fichier objet. Un Makefile doit permettre de compiler aisément le projet et un (bref) manuel d'utilisation doit être disponible (fichier `README.txt` par exemple).
- Un rapport de projet décrivant le protocole de communication utilisé et l'implémentation correspondante. Discutez aussi des cas problématiques et précisez clairement ce qui a été fait et ce qu'il ne l'a pas été.

Description :

Le but du projet est d'implémenter un serveur de fichiers distribué, analogue à ceux des premiers réseaux paire à paire (type Napster) : la machine de chaque utilisateur enregistré contient un ensemble de fichiers téléchargeables, et un unique serveur centralise les informations sur les fichiers disponibles. Après avoir récupéré les informations adéquates, les clients communiquent directement entre eux pour l'échange de fichiers.

Le langage de programmation sera soit le C (multi-processus), soit le Java (multi-threads). Au niveau des sockets, le choix entre un mode connecté et un mode datagramme est laissé libre pour chaque type de liaison, mais devra être commenté.

Version initiale :

Dans un premier temps chaque fichier est identifié par son nom.

Le serveur gère une liste de tous les fichiers disponibles et de toutes les machines possédant chaque fichier. Pour récupérer un fichier distant, un client doit d'abord se connecter au serveur et lui comuniquer la liste de ses propres fichiers.

Il peut ensuite récuperer auprès du serveur, et pour chaque fichier demandé, l'adresse des paires possédant ce fichier (un même fichier peut être présent sur plusieurs machines) et il se connecte alors directement à une unique paire. Une interface homme-machine sommaire devra permettre cette opération.

Dans cette version initiale, le serveur n'aura qu'un seul thread (ou processus) et traitera les requêtes de façon séquentielle.

Que se passe-t-il lorsqu'une paire souhaite se déconnecter du réseau ? Mettez en place une solution au niveau du serveur et éventuellement des autres paires.

Améliorations (facultatives) :

Dans un deuxième temps, vous devrez mettre en place les améliorations suivantes :

- un identifiant de fichier unique qui permette de distinguer deux fichiers avec le même nom, mais un contenu différent (voir par exemple la commande “md5sum”, qui vous permettra aussi de vérifier que le téléchargement s’est bien déroulé . . .),
- une mise à jour dynamique de la liste des fichiers sur le serveur : cette liste sera mise à jour en fonction des téléchargements effectués par chaque paire,
- une mise en place d’un serveur multi-threads ou multi-processus : attention aux synchronisations entre threads (ou processus).

Extension (possible) :

Voici une extension possible :

- un téléchargement des fichiers par blocs : un client doit pouvoir récupérer les différents blocs d’un même fichier sur plusieurs clients à la fois.